Remarks

The present amendment responds to the Official Action dated July 6, 2004. The Official Action objected to claim 14 as informal. Claims 6-8, 13-15, and 17-22 were rejected under 35 U.S.C. §102 (e) based on North et al. U.S. Patent No. 6,055,619 (North). In paragraphs 10-14 of the Official Action, claims 18-22 were rejected under 35 U.S.C. §102 (e) based on Kerr et al. U.S. Patent No. 6,105,119 (Kerr).

Claims 6, 8,13-15, 17, 19-22 have been amended to be more clear and distinct. Claim 23 has been newly added to more completely cover aspects of the slot function short instruction words (SFSIWs) associated with the LV2 instruction. Claims 24 and 25 have been newly added to more completely cover another aspect of the present invention for loading a partitioned VIM with instructions stored by DMA operations in a local data memory. Claims 6-8, 13-15, and 17-25 are presently pending.

Typographical Error in the Specification

During review of the specification, a typographical error was found in the paragraph beginning at page 14, line 21. The typo concerns the loading of the 16th instruction. The equation page 15, line 6:

(V[01]+VIMOFFS + InstrCnt)[UnitVIM]←(InstrCnt)th Instruction following LV2

has been amended to:

(V[01]+VIMOFFS + InstrCnt)[UnitVIM]←(InstrCnt<u>+ 1</u>)th Instruction following LV2

to correctly correspond to the text at page 15, lines 7 and 8 which correctly states: "InstrCnt is a binary coded number, 0 thru F, that represents from 1 to 16 instructions that can be loaded into up to 16 consecutive UnitVIM locations." For example, for the 16^{th} instruction to be loaded, InstrCnt would be set to a maximum setting of F, a binary setting of all 1's in the four bit InstrCnt bit field of bits 18-21 in LV2 instruction 450 Fig. 4C, and the equation page 15, line 6 would then indicate, by substituting F for InstrCnt, as follows:

(V[01]+VIMOFFS + F)[UnitVIM]←(F + 1)th Instruction following LV2

which equivalently in decimal is:

(V[01]+VIMOFFS + 15)[UnitVIM]←(16)th Instruction following LV2

Consequently, with (V[01]+VIMOFFS)=b as the start address where the first instruction is loaded, sixteen VIM addresses b to b+15 are loaded, where b+15 is the 16^{th} instruction loaded.

<u>Informality Objection to Claim 14</u>

Claim 14 has been amended to correct the noted misspelling of the word port.

<u>The Art Rejections</u>

As addressed in greater detail below, North and Kerr do not support the Official Action's reading of them and the rejections based thereupon should be reconsidered and withdrawn. Further, the Applicants do not acquiesce in the analysis of North and Kerr made by the Official Action and respectfully traverse the Official Action's analysis underlying its rejections.

Claims 6-8, 13-15, and 17-22 were indicated to be rejected under 35 U.S.C. §102(e) based on North, at page 2, paragraph 2 of the Official Action. No analysis of claims 18-22 was

made with respect to North and as these claims were separately addressed at length in paragraphs 10-14 of the Official Action based on Kerr, it is assumed the rejection based on North applies only to claims 6-8, 13-15, and 17.

North discloses direct memory access circuitry 208 for controlling direct memory accesses to a selected program memory 202 and a data memory 203/204. North Abstract. Referring to Figs. 1 and 2a of North upon which the Official Action relies, a stream processor 100, part of accelerator 200, processes multiple streams of audio data. North, col. 12, lines 27-46. To this end, the stream processor 100 includes a local register file 101 with registers accessible either as individual 16-bit registers or as pairs of 32-bit registers. North, col. 8, lines 15-18. However, North is silent with respect to associating an individual instruction storage unit with the arithmetic logic unit 104 and associating an individual instruction storage unit with the multiply accumulate unit 105, where the individual instruction storage units are separate from the program memory 202. The stream processor core 100 of North Fig. 1 does not illustrate any instruction storage directly associated with the arithmetic logic unit 104 or the multiply accumulate unit 105 of the core 100. As illustrated in Fig. 2A, DMA transfers can only occur between random access memories (RAMs), such as Program Memory 202, Parameter RAM 203, and Sample RAM 204 connected on the 32-bit DMA bus 209 or between a PCI attached device and the 32-bit DMA bus 209 attached RAMs. North, Fig. 2A and col. 9, lines 3-26. The RAMs 202, 203, and 204 are not special purpose RAMs but are standard RAMs that store program instructions or data. No direct DMA bus connection to the stream processor core 100 is shown or described.

In contrast, the present invention's ManArray processor core 100 of Fig. 1 has co-resident special purpose very long instruction word (VLIW) memories (VIMs) that are directly DMA accessible, such as VIM 109 in SP/PE0 101. The DMA controller 181 retrieves short instruction words (SIWs) from external DMA memory to be included in a DMA packet and then sent over a ManArray data bus 183 to store in local VIMs on each processing element, for example VIM 109, over internal DMA interfaces 173, 175, 177, and 179. See page 8, lines 16-19 of the present specification. The DMA interfaces advantageously provide SIWs to the local VIMs to create VLIWs in VIM storage using the DMA data packet. Once a VLIW is stored in the VIMs, it is accessible for local execution by using a program execute VLIW instruction. The VIMs are special purpose memories that are separate from and in addition to the processor's program memory and are directly associated with the processor's execution units.

Referring to Figs. 6, 8, and 9, the VIM is partitioned into separate VIM sections each associated with a slot function SIW (SFSIW) and the associated functional execute unit. For example, VIM 616 and VIM 808 consist of multiple independent memory units each associated with their functional execute units. See page 13, lines 7-9, page 14, lines 12-13, and page 17, lines 15-16. For example, one separate VIM section stores arithmetic logic unit (ALU) instructions while another separate VIM section may store multiply-accumulate (MAU) instructions. In order to load VLIWs into a VIM comprised of separate VIM sections, a VIM load controller receives SIWs from individual DMA data packets and loads each separate VIM section independently of the others. Each DMA data packet is associated with a separate VIM section.

Claim 6, as presently amended, recites

> A very long instruction word (VLIW) memory (VIM) direct memory access (DMA) apparatus comprising:
> a plurality of VLIW slot function units;
> a partitioned VLIW memory (VIM) having a separate VIM section per VLIW slot function unit, each separate VIM section having a plurality of memory locations for storing slot function unit instructions;
> a DMA interface receiving a data packet having a plurality of short instruction Words (SIWs); and
> a VIM load controller for selecting a separate VIM section and selectively controlling the loading of at least one SIW from the received plurality of SIWs into the selected separate VIM section. (emphasis added)

See also claims 13 and 17.

North does not claim and does not disclose "a partitioned VLIW memory (VIM) having a separate VIM section per VLIW slot function unit, each separate VIM section having a plurality of memory locations for storing slot function unit instructions" as presently claimed. North does not claim and does not disclose "a VIM load controller for selecting a separate VIM section and selectively controlling the loading of at least one SIW from the received plurality of SIWs into the selected separate VIM section" as presently claimed in claim 6. North simply loads RAMs external to the stream processor core with program instructions and data retrieved by a DMA controller without associating separate VIM sections with a VLIW slot function unit as presently claimed.

Claims 18-22 were presumably rejected under 35 U.S.C. §102(e) based on Kerr. At col. 141, lines 52-65, Kerr describes a VLIW digital signal processor(DSP) integrated circuit 2110. Referring to Fig. 21, the DSP has two sets of processing blocks including single precision/double precision integer multiply unit, arithmetic logic unit branch unit, arithmetic logic unit bit counter

unit, and an add/subtract address/load store unit. Like North, Kerr's processing blocks couple to a local register file 2150. The disclosure in Kerr is silent with respect to associating separate and individual memories with the execution units of the VLIW DSP 2110, such as M1, M2 (single precision/double precision (SP/DP) integer multiply (INT MPY), S1, S2 ALU shifter, bitfield branch, L1, L2 (SP/DP) (INT ALU) bit counter, and D1, D2 ADD/SUB address load-store units. The VLIW DSP 2110 of Fig. 21 does not illustrate any instruction storage directly associated with the execution units M1, M2, S1, S2, L1, L2, D1, and D2. An internal bus, DSP internal bus 2140, couples the register file 2150 to local RAM, such as 2120 and a DSP interface 2130. No explicit or implied DMA path is illustrated or discussed that could be used to transfer instructions to the DSP where the transferred instructions could be separately stored with their associated execution units.

In contrast to Kerr, the present invention uses a VLIW memory divided up into separate VIM sections where each separate VIM section is associated with an execution unit. Individual short instruction words are transferred by DMA data packets to the separate VIM sections to store the instructions as pending instructions to be accessed by a program execute VLIW instruction for execution. Kerr, like North, does not claim and does not disclose "a VLIW memory (VIM) having a plurality of VIM sections, each VIM section storing short instruction words (SIWs) corresponding to one of the plurality of functional execution units, each VIM section associated with one of the plurality of functional execution units, each stored SIW being selectable from any VIM section for parallel execution with any other stored SIW associated with a different functional execution unit of the plurality functional execution units" as presently

claimed in claim 17. Kerr, like North, does not claim and does not disclose "a VIM load controller for separately controlling the loading of at least one SIW of the received plurality of SIWs into a separate VIM section" as presently claimed in claim 17. Kerr as well as North simply loads RAMs external to the processors with program instructions and data retrieved by a DMA controller.

The Official Action relies on Fig. 1 of Kerr to suggest use of a "plurality of VIM sections". The Applicants respectfully disagree. Fig. 1 of Kerr is a "block diagram of improved integrated circuits and computer system embodiments for desktop and mobile computers, television sets, set-top boxes and appliances improved with asymmetrical multiprocessors." Kerr, col. 3, lines 58-61. This is clearly different from the present claimed function.

New claim 24 has been added to cover the aspect of loading a separate VIM section from local data memory storing a DMA data packet containing SIWs. The loading of the separate VIM section is through the use of a VIM load instruction. Claim 24 recites "a local data memory for storing a DMA data packet having at least one SIW associated with a VIM section; and a VIM load instruction specifying an operation that causes at least one DMA data packet SIW to be read from the local data memory and be loaded in a specified VIM section." Kerr and North, taken separately or in combination, do not disclose or make obvious the features as claimed in claim 24.

In summary, the relied upon art does not indicate a recognition of the problems addressed by the present invention. Further, the relied upon art does not teach and does not suggest an apparatus which would solve the problems of DMA loading of multiple separate VIM section

memories co-located with a core processor as addressed by the present invention in the manner solved by the present invention. The claims as presently amended are not taught, are not inherent, and are not obvious in light of the relied upon art.

Conclusion

All of the presently pending claims, as amended, appearing to define over the applied references, withdrawal of the present rejection and prompt allowance are requested.

Respectfully submitted,

Joseph B. Agusta
Reg. No. 52,547
Priest & Goldstein, PLLC
5015 Southpark Drive, Suite 230
Durham, NC 27713-7736
(919) 806-1600